```
   AAAAAAAAA     NNN         NNN     AAAAAAAAA     LLL               YYY         YYY  ZZZZZZZZZZZZZZZ
   AAAAAAAAA     NNN         NNN     AAAAAAAAA     LLL               YYY         YYY  ZZZZZZZZZZZZZZZ
   AAAAAAAAA     NNN         NNN     AAAAAAAAA     LLL               YYY         YYY  ZZZZZZZZZZZZZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL               YYY         YYY              ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL               YYY         YYY              ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL               YYY         YYY              ZZZ
AAA         AAA  NNNNNN      NNN  AAA         AAA  LLL                  YYY   YYY              ZZZ
AAA         AAA  NNNNNN      NNN  AAA         AAA  LLL                  YYY   YYY              ZZZ
AAA         AAA  NNNNNN      NNN  AAA         AAA  LLL                  YYY   YYY              ZZZ
AAA         AAA  NNN   NNN   NNN  AAA         AAA  LLL                     YYY              ZZZ
AAA         AAA  NNN   NNN   NNN  AAA         AAA  LLL                     YYY              ZZZ
AAA         AAA  NNN   NNN   NNN  AAA         AAA  LLL                     YYY              ZZZ
AAAAAAAAAAAAAAA  NNN      NNNNNN  AAAAAAAAAAAAAAA  LLL                     YYY           ZZZ
AAAAAAAAAAAAAAA  NNN      NNNNNN  AAAAAAAAAAAAAAA  LLL                     YYY           ZZZ
AAAAAAAAAAAAAAA  NNN      NNNNNN  AAAAAAAAAAAAAAA  LLL                     YYY           ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL                     YYY        ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL                     YYY        ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL                     YYY        ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLLLLLLLLLLLLLL         YYY        ZZZZZZZZZZZZZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLLLLLLLLLLLLLL         YYY        ZZZZZZZZZZZZZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLLLLLLLLLLLLLL         YYY        ZZZZZZZZZZZZZZZ
```

```
**FILE**ID**OBJEXEOUT
```

```
  000000    BBBBBBBB            JJ  EEEEEEEEEE  XX      XX  EEEEEEEEEE    000000    UU      UU  TTTTTTTTTT
  000000    BBBBBBBB            JJ  EEEEEEEEEE  XX      XX  EEEEEEEEEE    000000    UU      UU  TTTTTTTTTT
00      00  BB      BB          JJ  EE          XX      XX  EE          00      00  UU      UU      TT
00      00  BB      BB          JJ  EE          XX      XX  EE          00      00  UU      UU      TT
00      00  BB      BB          JJ  EE            XX  XX    EE          00      00  UU      UU      TT
00      00  BB      BB          JJ  EE            XX  XX    EE          00      00  UU      UU      TT
00      00  BBBBBBBB            JJ  EEEEEEEE        XX      EEEEEEEE    00      00  UU      UU      TT
00      00  BBBBBBBB            JJ  EEEEEEEE        XX      EEEEEEEE    00      00  UU      UU      TT
00      00  BB      BB  JJ      JJ  EE            XX  XX    EE          00      00  UU      UU      TT
00      00  BB      BB  JJ      JJ  EE            XX  XX    EE          00      00  UU      UU      TT
00      00  BB      BB  JJ      JJ  EE          XX      XX  EE          00      00  UU      UU      TT
00      00  BB      BB  JJ      JJ  EE          XX      XX  EE          00      00  UU      UU      TT      ....
  000000    BBBBBBBB      JJJJJJ    EEEEEEEEEE  XX      XX  EEEEEEEEEE    000000    UUUUUUUUUU      TT      ....
  000000    BBBBBBBB      JJJJJJ    EEEEEEEEEE  XX      XX  EEEEEEEEEE    000000    UUUUUUUUUU      TT      ....
                                                                                                            ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0 %title 'OBJEXEOUT - Handle Report Output'
0       module objexeout(
1                       ident='V04-000') = begin
1 !
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! Facility:     VAX/VMS Analyze Facility, Handle Report Output
1 !
1 ! Abstract:     This module is responsible for generating report output
1 !               for ANALYZE/OBJECT and ANALYZE/IMAGE.  It provides the
1 !               capability to create report files and fill them with
1 !               output lines.
1 !
1 !
1 ! Environment:
1 !
1 ! Author: Paul C. Anagnostopoulos, Creation Date: 8 January 1981
1 !
1 ! Modified By:
1 !
1 !       V03-005 DGB0067         Donald G. Blair         03-Jul-1984
1 !               Support the /NOOUTPUT qualifier.
1 !
1 !       V03-004 DGB0053         Donald G. Blair         10-May-1984
1 !               When an error occurs, save the error status so
1 !               we can return it correctly at image exit.
1 !
1 !       V03-003 PCA1011         Paul C. Anagnostopoulos  1-Apr-1983
1 !               Change the message prefix to ANLOBJ$_ to ensure that
1 !               message symbols are unique across all ANALYZEs.  This
1 !               is necessitated by the new merged message files.
1 !
1 !       V03-002 PCA0021         Paul Anagnostopoulos    24-Mar-1982
```

```
;    58             0058  1 !               Signal errors using the correct STV values.
;    59             0059  1 !
;    60             0060  1 !       V03-001 PCA0015         Paul Anagnostopoulos    22-Mar-1982
;    61             0061  1 !               Don't constrain report file lines to 132 characters.
;    62             0062  1 !--
```

```
%sbttl 'Module Declarations'
!
! Libraries and Requires:
!

library 'starlet';
require 'objexereq';

!
! Table of Contents:
!

forward routine
        anl$prepare_report_file: novalue,
        anl$report_page: novalue,
        anl$report_line: novalue,
        anl$exit_with_status: novalue,
        anl$format_line: novalue,
        anl$format_error: novalue,
        anl$error_count: novalue,
        anl$format_hex: novalue,
        anl$format_flags: novalue,
        anl$format_data_type: novalue,
        anl$format_mask: novalue,
        anl$format_protection: novalue,
        anl$format_severity: novalue,
        anl$interact;

!
! External References:
!

external routine
        cli$get_value: addressing_mode(general),
        lib$get_input: addressing_mode(general),
        lib$lp_lines: addressing_mode(general),
        cli$present: addressing_mode(general),
        str$trim: addressing_mode(general);

external
        anl$gb_interactive: byte;

!
! Global Variables
!

global
        anl$worst_error:            ! This contains either success status, or
            initial(anlobj$_ok);    ! if errors occur it contains the first error
                                    ! of the worst severity that occurred.
                                    ! This status is returned at image exit.

!
! Own Variables:
!
! The following data structures are needed to create and print to the
! report file.  They include the FAB and RAB, and a buffer for the report
```

```
;  121          0555  1 ! spec.
;  122          0556  1
;  123          0557  1 own
;  124          0558  1         own_described_buffer(report_spec,nam$c_maxrss),
;  125          0559  1
;  126        P 0560  1         report_fab: $fab(dnm='ANALYZE.ANL',
;  127        P 0561  1                         fac=put,
;  128        P 0562  1                         fna=report_spec+8,
;  129        P 0563  1                         fns=nam$c_maxrss,
;  130        P 0564  1                         fop=sqo,
;  131        P 0565  1                         org=seq,
;  132        P 0566  1                         rat=cr,
;  133          0567  1                         rfm=var),
;  134          0568  1
;  135        P 0569  1         report_rab: $rab(fab=report_fab,
;  136          0570  1                         rac=seq);
;  137          0571  1
;  138          0572  1 ! The following variables are needed to format the report.
;  139          0573  1
;  140          0574  1 own
;  141          0575  1         generating_report,
;  142          0576  1         own_described_buffer(input_file_spec,nam$c_maxrss),
;  143          0577  1         report_heading_msg: long,
;  144          0578  1         page_number: long,
;  145          0579  1         line_counter: signed long;
;  146          0580  1
;  147          0581  1 ! We also need to keep track of how many errors were reported.
;  148          0582  1
;  149          0583  1 own
;  150          0584  1         error_count: long initial(0);
```

```
1 %sbttl 'ANL$PREPARE_REPORT_FILE - Prepare Report File'
1 !++
1 ! Functional Description:
1 !       This routine is called whenever we begin the analysis of a new
1 !       file.  It sets up a report file to receive the analysis.
1 !
1 ! Formal Parameters:
1 !       output_spec     The report file spec as specified by the user.
1 !                       This is used on the first call to create the file.
1 !       input_spec      The spec of the input file we are analyzing.
1 !       heading_msg     An optional message code specifying the report
1 !                       page heading.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$prepare_report_file(output_spec,input_spec,heading_msg): novalue = begin
2
2 bind
2       output_spec_dsc = .output_spec: descriptor,
2       input_spec_dsc = .input_spec: descriptor;
2
2 local
2       status: long;
2
2 builtin
2       nullparameter;
2
2
2 ! Are we generating a report?
2
2 generating_report = cli$present(describe('OUTPUT'));
2
2 ! If the report file is not open, then we want to create it and prepare
2 ! for the report.
2
3 if (.report_rab[rab$w_isi] eqlu 0) and .generating_report then (
3
3       ! Save the output file spec as the principal name of the report file.
3
3       ch$copy(.output_spec_dsc[len],.output_spec_dsc[ptr],
3               ' ',.report_spec[len],.report_spec[ptr]);
3       str$trim(report_spec,report_spec,report_spec);
3
3       ! Now let's create the report file and connect it.
3
;  209           0642 3            status = $create(fab=report_fab);
;  210           0643 3            check (.status, anlobj$_openout,1,report_spec,.status,.report_fab[fab$l_stv]);
;  211           0644 3            status = $connect(rab=report_rab);
;  212           0645 3            check (.status, .status);
;  213           0646 2 );
;  214           0647 2
;  215           0648 2 ! Now let's save the report heading message and the input file spec for
;  216           0649 2 ! a subheading.
;  217           0650 2
;  218           0651 2 report_heading_msg = .heading_msg;
;  219           0652 2 input_file_spec[len] = .input_spec_dsc[len];
;  220           0653 2 ch$copy(.input_spec_dsc[len],.input_spec_dsc[ptr],
;  221           0654 2          ' ',.input_file_spec[len],.input_file_spec[ptr]);
;  222           0655 2
;  223           0656 2 ! Now reset the page counter and start a new page.
;  224           0657 2
;  225           0658 2 page_number = 0;
;  226           0659 2 anl$report_page();
;  227           0660 2
;  228           0661 2 return;
;  229           0662 2
;  230           0663 1 end;


                                                                  .TITLE  OBJEXEOUT OBJEXEOUT - Handle Report Output
                                                                  .IDENT  \V04-000\

                                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2

         4C  4E  41  2E  45  5A  59  4C  41  4E  41  00000 P.AAA:  .ASCII  \ANALYZE.ANL\                              ;
                             54  55  50  54  55  4F  0000B P.AAC:  .ASCII  \OUTPUT\                                   ;
                                                     00011         .BLKB   3
                                           00000006  00014 P.AAB:  .LONG   6                                          ;
                                           00000000' 00018         .ADDRESS P.AAC                                     ;

                                                                  .PSECT  $OWN$,NOEXE,2

                                           000000FF  00000 REPORT_SPEC:
                                                                  .LONG   255                                        ;
                                           00000000' 00004         .ADDRESS REPORT_SPEC+8                             ;
                                                     00008         .BLKB   255
                                                     00107         .BLKB   1
                                                 03  00108 REPORT_FAB:
                                                                  .BYTE   3                                          ;
                                                 50  00109         .BYTE   80                                         ;
                                               0000  0010A         .WORD   0                                          ;
                                           00000040  0010C         .LONG   64                                         ;
                                           00000000  00110         .LONG   0                                          ;
                                           00000000  00114         .LONG   0                                          ;
                                           00000000  00118         .LONG   0                                          ;
                                               0000  0011C         .WORD   0                                          ;
                                                 01  0011E         .BYTE   1                                          ;
                                                 00  0011F         .BYTE   0                                          ;
                                           00000000  00120         .LONG   0                                          ;
                                                 00  00124         .BYTE   0                                          ;
                                                 00  00125         .BYTE   0                                          ;
                                                 02  00126         .BYTE   2                                          ;
```

```
                02  00127               .BYTE    2                                  ;
          00000000  00128               .LONG    0                                  ;
          00000000  0012C               .LONG    0                                  ;
          00000000  00130               .LONG    0                                  ;
         00000000'  00134               .ADDRESS REPORT_SPEC+8                      ;
         00000000'  00138               .ADDRESS P.AAA                              ;
                FF  0013C               .BYTE    -1                                 ;
                0B  0013D               .BYTE    11                                 ;
              0000  0013E               .WORD    0                                  ;
          00000000  00140               .LONG    0                                  ;
              0000  00144               .WORD    0                                  ;
                00  00146               .BYTE    0                                  ;
                00  00147               .BYTE    0                                  ;
          00000000  00148               .LONG    0                                  ;
          00000000  0014C               .LONG    0                                  ;
              0000  00150               .WORD    0                                  ;
                00  00152               .BYTE    0                                  ;
                00  00153               .BYTE    0                                  ;
          00000000  00154               .LONG    0                                  ;
                01  00158 REPORT_RAB:
                                        .BYTE    1                                  ;
                44  00159               .BYTE    68                                 ;
              0000  0015A               .WORD    0                                  ;
          00000000  0015C               .LONG    0                                  ;
          00000000  00160               .LONG    0                                  ;
          00000000  00164               .LONG    0                                  ;
             0000#  00168               .WORD    0[3]                               ;
              0000  0016E               .WORD    0                                  ;
          00000000  00170               .LONG    0                                  ;
              0000  00174               .WORD    0                                  ;
                00  00176               .BYTE    0                                  ;
                00  00177               .BYTE    0                                  ;
              0000  00178               .WORD    0                                  ;
              0000  0017A               .WORD    0                                  ;
          00000000  0017C               .LONG    0                                  ;
          00000000  00180               .LONG    0                                  ;
          00000000  00184               .LONG    0                                  ;
          00000000  00188               .LONG    0                                  ;
                00  0018C               .BYTE    0                                  ;
                00  0018D               .BYTE    0                                  ;
                00  0018E               .BYTE    0                                  ;
                00  0018F               .BYTE    0                                  ;
          00000000  00190               .LONG    0                                  ;
         00000000'  00194               .ADDRESS REPORT_FAB                         ;
          00000000  00198               .LONG    0                                  ;
                    0019C GENERATING_REPORT:
                                        .BLKB    4
          000000FF  001A0 INPUT_FILE_SPEC:
                                        .LONG    255                                ;
         00000000'  001A4               .ADDRESS INPUT_FILE_SPEC+8                  ;
                    001A8               .BLKB    255
                    002A7               .BLKB    1
                    002A8 REPORT_HEADING_MSG:
                                        .BLKB    4
                    002AC PAGE_NUMBER:
                                        .BLKB    4
                    002B0 LINE_COUNTER:
```

```
                                            .BLKB    4
                    00000000  002B4 ERROR_COUNT:
                                            .LONG    0                                                     ;

                                            .PSECT   $GLOBAL$,NOEXE,2

                    0000000G 00000 ANL$WORST_ERROR::
                                            .LONG    ANLOBJ$_OK                                            ;

                                            .EXTRN   ANLOBJ$_OK, ANLOBJ$_ANYTHING
                                            .EXTRN   ANLOBJ$_DATATYPE
                                            .EXTRN   ANLOBJ$_ERRORCOUNT
                                            .EXTRN   ANLOBJ$_ERRORNONE
                                            .EXTRN   ANLOBJ$_ERRORS, ANLOBJ$_EXEFIXA
                                            .EXTRN   ANLOBJ$_EXEFIXAIMAGE
                                            .EXTRN   ANLOBJ$_EXEFIXALINE
                                            .EXTRN   ANLOBJ$_EXEFIXCOUNT
                                            .EXTRN   ANLOBJ$_EXEFIXEXTRA
                                            .EXTRN   ANLOBJ$_EXEFIXFIXED
                                            .EXTRN   ANLOBJ$_EXEFIXFLAGS
                                            .EXTRN   ANLOBJ$_EXEFIXG
                                            .EXTRN   ANLOBJ$_EXEFIXGIMAGE
                                            .EXTRN   ANLOBJ$_EXEFIXGLINE
                                            .EXTRN   ANLOBJ$_EXEFIXLIST
                                            .EXTRN   ANLOBJ$_EXEFIXNAME
                                            .EXTRN   ANLOBJ$_EXEFIXNAMEO
                                            .EXTRN   ANLOBJ$_EXEFIXP
                                            .EXTRN   ANLOBJ$_EXEFIXPSECT
                                            .EXTRN   ANLOBJ$_EXEFIXUP
                                            .EXTRN   ANLOBJ$_EXEFIXUPNONE
                                            .EXTRN   ANLOBJ$_EXEGST, ANLOBJ$_EXEHDR
                                            .EXTRN   ANLOBJ$_EXEHDRACTIVE
                                            .EXTRN   ANLOBJ$_EXEHDRBLKCOUNT
                                            .EXTRN   ANLOBJ$_EXEHDRCHANCOUNT
                                            .EXTRN   ANLOBJ$_EXEHDRCHANDEF
                                            .EXTRN   ANLOBJ$_EXEHDRDECECO
                                            .EXTRN   ANLOBJ$_EXEHDRDMT
                                            .EXTRN   ANLOBJ$_EXEHDRDST
                                            .EXTRN   ANLOBJ$_EXEHDRFILEID
                                            .EXTRN   ANLOBJ$_EXEHDRFIXED
                                            .EXTRN   ANLOBJ$_EXEHDRFLAGS
                                            .EXTRN   ANLOBJ$_EXEHDRGBLIDENT
                                            .EXTRN   ANLOBJ$_EXEHDRGST
                                            .EXTRN   ANLOBJ$_EXEHDRIDENT
                                            .EXTRN   ANLOBJ$_EXEHDRIMAGEID
                                            .EXTRN   ANLOBJ$_EXEHDRISD
                                            .EXTRN   ANLOBJ$_EXEHDRISDBASE
                                            .EXTRN   ANLOBJ$_EXEHDRISDCOUNT
                                            .EXTRN   ANLOBJ$_EXEHDRISDFLAGS
                                            .EXTRN   ANLOBJ$_EXEHDRISDGBLNAM
                                            .EXTRN   ANLOBJ$_EXEHDRISDNUM
                                            .EXTRN   ANLOBJ$_EXEHDRISDPFCDEF
                                            .EXTRN   ANLOBJ$_EXEHDRISDPFCSIZ
                                            .EXTRN   ANLOBJ$_EXEHDRISDTYPE
                                            .EXTRN   ANLOBJ$_EXEHDRISDVBN
                                            .EXTRN   ANLOBJ$_EXEHDRLINKID
                                            .EXTRN   ANLOBJ$_EXEHDRMATCH
```

```
        .EXTRN  ANLOBJ$_EXEHDRNAME
        .EXTRN  ANLOBJ$_EXEHDRNOPATCH
        .EXTRN  ANLOBJ$_EXEHDRPAGECOUNT
        .EXTRN  ANLOBJ$_EXEHDRPAGEDEF
        .EXTRN  ANLOBJ$_EXEHDRPATCH
        .EXTRN  ANLOBJ$_EXEHDRPATCHDATE
        .EXTRN  ANLOBJ$_EXEHDRPRIV
        .EXTRN  ANLOBJ$_EXEHDRROPATCH
        .EXTRN  ANLOBJ$_EXEHDRRWPATCH
        .EXTRN  ANLOBJ$_EXEHDRSYMDBG
        .EXTRN  ANLOBJ$_EXEHDRSYSVER
        .EXTRN  ANLOBJ$_EXEHDRTEXTVBN
        .EXTRN  ANLOBJ$_EXEHDRTIME
        .EXTRN  ANLOBJ$_EXEHDRTYPEEXE
        .EXTRN  ANLOBJ$_EXEHDRTYPELIM
        .EXTRN  ANLOBJ$_EXEHDRUSERECO
        .EXTRN  ANLOBJ$_EXEHDRXFER1
        .EXTRN  ANLOBJ$_EXEHDRXFER2
        .EXTRN  ANLOBJ$_EXEHDRXFER3
        .EXTRN  ANLOBJ$_EXEHEADING
        .EXTRN  ANLOBJ$_EXEPATCH
        .EXTRN  ANLOBJ$_FLAG, ANLOBJ$_HEXDATA
        .EXTRN  ANLOBJ$_HEXHEADING1
        .EXTRN  ANLOBJ$_HEXHEADING2
        .EXTRN  ANLOBJ$_INDMSGSEC
        .EXTRN  ANLOBJ$_INTERACT
        .EXTRN  ANLOBJ$_MASK, ANLOBJ$_OBJCPRREC
        .EXTRN  ANLOBJ$_OBJDBGREC
        .EXTRN  ANLOBJ$_OBJENV, ANLOBJ$_OBJEOMFLAGS
        .EXTRN  ANLOBJ$_OBJEOMREC
        .EXTRN  ANLOBJ$_OBJEOMSEVABT
        .EXTRN  ANLOBJ$_OBJEOMSEVERR
        .EXTRN  ANLOBJ$_OBJEOMSEVIGN
        .EXTRN  ANLOBJ$_OBJEOMSEVRES
        .EXTRN  ANLOBJ$_OBJEOMSEVSUC
        .EXTRN  ANLOBJ$_OBJEOMSEVWRN
        .EXTRN  ANLOBJ$_OBJEOMWREC
        .EXTRN  ANLOBJ$_OBJFADPASSMECH
        .EXTRN  ANLOBJ$_OBJGSDENV
        .EXTRN  ANLOBJ$_OBJGSDENVFLAGS
        .EXTRN  ANLOBJ$_OBJGSDENVPAR
        .EXTRN  ANLOBJ$_OBJGSDEPM
        .EXTRN  ANLOBJ$_OBJGSDEPMW
        .EXTRN  ANLOBJ$_OBJGSDIDC
        .EXTRN  ANLOBJ$_OBJGSDIDCENT
        .EXTRN  ANLOBJ$_OBJGSDIDCFLAGS
        .EXTRN  ANLOBJ$_OBJGSDIDCMATCH
        .EXTRN  ANLOBJ$_OBJGSDIDCOBJ
        .EXTRN  ANLOBJ$_OBJGSDIDCVALA
        .EXTRN  ANLOBJ$_OBJGSDIDCVALB
        .EXTRN  ANLOBJ$_OBJGSDLEPM
        .EXTRN  ANLOBJ$_OBJGSDLPRO
        .EXTRN  ANLOBJ$_OBJGSDLSY
        .EXTRN  ANLOBJ$_OBJGSDPRO
        .EXTRN  ANLOBJ$_OBJGSDPROW
        .EXTRN  ANLOBJ$_OBJGSDPSC
        .EXTRN  ANLOBJ$_OBJGSDPSCALIGN
```

```
                                                              .EXTRN  ANLOBJ$_OBJGSDPSCALLOC
                                                              .EXTRN  ANLOBJ$_OBJGSDPSCBASE
                                                              .EXTRN  ANLOBJ$_OBJGSDPSCFLAGS
                                                              .EXTRN  ANLOBJ$_OBJGSDREC
                                                              .EXTRN  ANLOBJ$_OBJGSDSPSC
                                                              .EXTRN  ANLOBJ$_OBJGSDSYM
                                                              .EXTRN  ANLOBJ$_OBJGSDSYMW
                                                              .EXTRN  ANLOBJ$_OBJGTXREC
                                                              .EXTRN  ANLOBJ$_OBJHDRIGNREC
                                                              .EXTRN  ANLOBJ$_OBJHEADING
                                                              .EXTRN  ANLOBJ$_OBJLITINDEX
                                                              .EXTRN  ANLOBJ$_OBJLNKREC
                                                              .EXTRN  ANLOBJ$_OBJLNMREC
                                                              .EXTRN  ANLOBJ$_OBJMHDCREATE
                                                              .EXTRN  ANLOBJ$_OBJMHDNAME
                                                              .EXTRN  ANLOBJ$_OBJMHDPATCH
                                                              .EXTRN  ANLOBJ$_OBJMHDREC
                                                              .EXTRN  ANLOBJ$_OBJMHDRECSIZ
                                                              .EXTRN  ANLOBJ$_OBJMHDSTRLVL
                                                              .EXTRN  ANLOBJ$_OBJMHDVERSION
                                                              .EXTRN  ANLOBJ$_OBJMTCCORRECT
                                                              .EXTRN  ANLOBJ$_OBJMTCINPUT
                                                              .EXTRN  ANLOBJ$_OBJMTCNAME
                                                              .EXTRN  ANLOBJ$_OBJMTCREC
                                                              .EXTRN  ANLOBJ$_OBJMTCSEQNUM
                                                              .EXTRN  ANLOBJ$_OBJMTCUIC
                                                              .EXTRN  ANLOBJ$_OBJMTCVERSION
                                                              .EXTRN  ANLOBJ$_OBJMTCWHEN
                                                              .EXTRN  ANLOBJ$_OBJPROARGCOUNT
                                                              .EXTRN  ANLOBJ$_OBJPROARGNUM
                                                              .EXTRN  ANLOBJ$_OBJPSECT
                                                              .EXTRN  ANLOBJ$_OBJSRCREC
                                                              .EXTRN  ANLOBJ$_OBJSTATHEADING1
                                                              .EXTRN  ANLOBJ$_OBJSTATHEADING2
                                                              .EXTRN  ANLOBJ$_OBJSTATLINE
                                                              .EXTRN  ANLOBJ$_OBJSTATTOTAL
                                                              .EXTRN  ANLOBJ$_OBJSYMBOL
                                                              .EXTRN  ANLOBJ$_OBJSYMFLAGS
                                                              .EXTRN  ANLOBJ$_OBJTIRARGINDEX
                                                              .EXTRN  ANLOBJ$_OBJTIRCMD
                                                              .EXTRN  ANLOBJ$_OBJTIRCMDSTK
                                                              .EXTRN  ANLOBJ$_OBJTBTREC
                                                              .EXTRN  ANLOBJ$_OBJTIRREC
                                                              .EXTRN  ANLOBJ$_OBJTIRSTOIM
                                                              .EXTRN  ANLOBJ$_OBJTIRVIELD
                                                              .EXTRN  ANLOBJ$_OBJTTLREC
                                                              .EXTRN  ANLOBJ$_OBJVALUE
                                                              .EXTRN  ANLOBJ$_OBJUVALUE
                                                              .EXTRN  ANLOBJ$_PROTECTION
                                                              .EXTRN  ANLOBJ$_SEVERITY
                                                              .EXTRN  ANLOBJ$_TEXT, ANLOBJ$_TEXTHDR
                                                              .EXTRN  ANLOBJ$_NOSUCHMOD
                                                              .EXTRN  ANLOBJ$_BADDATE
                                                              .EXTRN  ANLOBJ$_BADHDRBLKCOUNT
                                                              .EXTRN  ANLOBJ$_BADSEVERITY
                                                              .EXTRN  ANLOBJ$_BADSYM1ST
                                                              .EXTRN  ANLOBJ$_BADSYMCHAR
```

```
                                        .EXTRN  ANLOBJ$_BADSYMLEN
                                        .EXTRN  ANLOBJ$_EXEBADFIXUPEND
                                        .EXTRN  ANLOBJ$_EXEBADFIXUPISD
                                        .EXTRN  ANLOBJ$_EXEBADFIXUPVBN
                                        .EXTRN  ANLOBJ$_EXEBADISDS1
                                        .EXTRN  ANLOBJ$_EXEBADISDTYPE
                                        .EXTRN  ANLOBJ$_EXEBADMATCH
                                        .EXTRN  ANLOBJ$_EXEBADPATCHLEN
                                        .EXTRN  ANLOBJ$_EXEBADOBJ
                                        .EXTRN  ANLOBJ$_EXEBADTYPE
                                        .EXTRN  ANLOBJ$_EXEBADXFER0
                                        .EXTRN  ANLOBJ$_EXEHDRISDLONG
                                        .EXTRN  ANLOBJ$_EXEHDRLONG
                                        .EXTRN  ANLOBJ$_EXEISDLENDZRO
                                        .EXTRN  ANLOBJ$_EXEISDLENGBL
                                        .EXTRN  ANLOBJ$_EXEISDLENPRIV
                                        .EXTRN  ANLOBJ$_EXENOTNATIVE
                                        .EXTRN  ANLOBJ$_EXTRABYTES
                                        .EXTRN  ANLOBJ$_FIELDFIT
                                        .EXTRN  ANLOBJ$_FLAGERROR
                                        .EXTRN  ANLOBJ$_NOTOK, ANLOBJ$_OBJBADIDCMATCH
                                        .EXTRN  ANLOBJ$_OBJBADNUM
                                        .EXTRN  ANLOBJ$_OBJBADPOP
                                        .EXTRN  ANLOBJ$_OBJBADPUSH
                                        .EXTRN  ANLOBJ$_OBJBADTYPE
                                        .EXTRN  ANLOBJ$_OBJBADVIELD
                                        .EXTRN  ANLOBJ$_OBJEOMBADSEV
                                        .EXTRN  ANLOBJ$_OBJEOMMISSING
                                        .EXTRN  ANLOBJ$_OBJFADBADAVC
                                        .EXTRN  ANLOBJ$_OBJFADBADRBC
                                        .EXTRN  ANLOBJ$_OBJGSDBADALIGN
                                        .EXTRN  ANLOBJ$_OBJGSDBADSUBTYP
                                        .EXTRN  ANLOBJ$_OBJHDRRES
                                        .EXTRN  ANLOBJ$_OBJMHDBADRECSIZ
                                        .EXTRN  ANLOBJ$_OBJMHDBADSTRLVL
                                        .EXTRN  ANLOBJ$_OBJMHDMISSING
                                        .EXTRN  ANLOBJ$_OBJNONTIRCMD
                                        .EXTRN  ANLOBJ$_OBJNOPSC
                                        .EXTRN  ANLOBJ$_OBJNULLREC
                                        .EXTRN  ANLOBJ$_OBJPOSPACE
                                        .EXTRN  ANLOBJ$_OBJPROMINMAX
                                        .EXTRN  ANLOBJ$_OBJPSCABSLEN
                                        .EXTRN  ANLOBJ$_OBJRECTOOBIG
                                        .EXTRN  ANLOBJ$_OBJTIRRES
                                        .EXTRN  ANLOBJ$_OBJUNDEFENV
                                        .EXTRN  ANLOBJ$_OBJUNDEFLIT
                                        .EXTRN  ANLOBJ$_OBJUNDEFPSC
                                        .EXTRN  ANALYZE$_FACILITY
                                        .EXTRN  CLI$GET_VALUE, LIB$GET_INPUT
                                        .EXTRN  LIB$LP_LINES, CLI$PRESENT
                                        .EXTRN  STR$TRIM, ANL$GB_INTERACTIVE
                                        .EXTRN  SYS$CREATE, SYS$CONNECT

                                        .PSECT  $CODE$,NOWRT,2

                        01FC 00000      .ENTRY  ANL$PREPARE_REPORT_FILE, Save R2,R3,R4,R5,- ; 0612
                                                R6,R7,R8                                      ;
```

```
                                 58 00000000G  00  9E 00002         MOVAB   LIB$SIGNAL, R8                                  ;
                                 57     0000'  CF  9E 00009         MOVAB   REPORT_SPEC, R7                                 ;
                                 52       04   AC  D0 0000E         MOVL    OUTPUT_SPEC, R2                                 ; 0615
                                 56       08   AC  D0 00012         MOVL    INPUT_SPEC, R6                                  ; 0616
                                        0000'  CF  9F 00016         PUSHAB  P.AAB                                           ; 0627
                       00000000G  00           01  FB 0001A         CALLS   #1, CLI$PRESENT                                 ;
                            019C  C7           50  D0 00021         MOVL    R0, GENERATING_REPORT                           ;
                                        015A   C7  B5 00026         TSTW    REPORT_RAB+2                                    ; 0632
                                               54  12 0002A         BNEQ    2$                                              ;
                                 4F     019C   C7  E9 0002C         BLBC    GENERATING_REPORT, 2$                           ;
       67            20       04   B2          62  2C 00031         MOVC5   (R2), @4(R2), #32, REPORT_SPEC, -               ; 0637
                                          04   B7     00037                 @REPORT_SPEC+4                                  ;
                                               57  DD 00039         PUSHL   R7                                              ; 0638
                                               57  DD 0003B         PUSHL   R7                                              ;
                                               57  DD 0003D         PUSHL   R7                                              ;
                       00000000G  00           03  FB 0003F         CALLS   #3, STR$TRIM                                    ;
                                        0108   C7  9F 00046         PUSHAB  REPORT_FAB                                      ; 0642
                       00000000G  00           01  FB 0004A         CALLS   #1, SYS$CREATE                                  ;
                                  52           50  D0 00051         MOVL    R0, STATUS                                      ;
                                  13           52  E8 00054         BLBS    STATUS, 1$                                      ; 0643
                                        0114   C7  DD 00057         PUSHL   REPORT_FAB+12                                   ;
                                               52  DD 0005B         PUSHL   STATUS                                          ;
                                               57  DD 0005D         PUSHL   R7                                              ;
                                               01  DD 0005F         PUSHL   #1                                              ;
                                    00B110A4   8F  DD 00061         PUSHL   #11604132                                       ;
                                  68           05  FB 00067         CALLS   #5, LIB$SIGNAL                                  ;
                                        0158   C7  9F 0006A  1$:    PUSHAB  REPORT_RAB                                      ; 0644
                       00000000G  00           01  FB 0006E         CALLS   #1, SYS$CONNECT                                 ;
                                  52           50  D0 00075         MOVL    R0, STATUS                                      ;
                                  05           52  E8 00078         BLBS    STATUS, 2$                                      ; 0645
                                               52  DD 0007B         PUSHL   STATUS                                          ;
                                  68           01  FB 0007D         CALLS   #1, LIB$SIGNAL                                  ;
                            02A8  C7      0C   AC  D0 00080  2$:    MOVL    HEADING_MSG, REPORT_HEADING_MSG                 ; 0651
                            01A0  C7           66  B0 00086         MOVW    (R6), INPUT_FILE_SPEC                           ; 0652
    01A0   C7        20       04  B6           66  2C 0008B         MOVC5   (R6), @4(R6), #32, INPUT_FILE_SPEC, -           ; 0654
                                        01A4   D7     00093                 @INPUT_FILE_SPEC+4                              ;
                                        02AC   C7  D4 00096         CLRL    PAGE_NUMBER                                     ; 0658
                           0000V  CF           00  FB 0009A         CALLS   #0, ANL$REPORT_PAGE                             ; 0659
                                               04     0009F         RET                                                     ; 0663

; Routine Size:  160 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 %sbttl 'ANL$REPORT_PAGE - Eject Page in Report'
1 !++
1 ! Functional Description:
1 !       This routine is called to eject the page in a report and print
1 !       the heading on the new page.
1 !
1 ! Formal Parameters:
1 !       none
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$report_page: novalue = begin
2
2
2 ! Since we are starting a new page, reset the line counter.
2
2 line_counter = lib$lp_lines() - 7;
2
2 ! If this is an interactive session, don't print any page headings.
2 ! They will really annoy the poor guy.
2
2 if .anl$gb_interactive then
2       return;
2
2 ! Eject the page.
2
2 anl$report_line(-1,describe(%char(formfeed)));
2
2 ! Increment the page number for the new page and print the heading lines.
2
2 increment (page_number);
2 anl$format_line(-1,0,.report_heading_msg,0,.page_number);
2 anl$format_line(-1,0,anlobj$_anything,input_file_spec);
2 anl$report_line(-1);
2 anl$report_line(-1);
2
2 return;
2
1 end;


                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

                                 0C  0001C P.AAE:  .ASCII  <12>
```

```
                                                0001D          .BLKB    3
                                    00000001  00020 P.AAD:    .LONG    1                                          ;
                                    00000000' 00024           .ADDRESS P.AAE                                      ;


                                                              .PSECT  $CODE$,NOWRT,2

                                         000C 00000           .ENTRY  ANL$REPORT_PAGE, Save R2,R3                ; 0687
                   53        0000V  CF    9E 00002           MOVAB   ANL$REPORT_LINE, R3                        ;
                   52        0000'  CF    9E 00007           MOVAB   PAGE_NUMBER, R2                            ;
       00000000G   00               00    FB 0000C           CALLS   #0, LIB$LP_LINES                           ; 0692
              04   A2           F9  A0    9E 00013           MOVAB   -7(R0), LINE_COUNTER                       ;
                   3D        0000G  CF    E8 00018           BLBS    ANL$GB_INTERACTIVE, 1$                     ; 0697
                             0000'  CF    9F 0001D           PUSHAB  P.AAD                                      ; 0702
                   7E               01    CE 00021           MNEGL   #1, -(SP)                                  ;
                   63               02    FB 00024           CALLS   #2, ANL$REPORT_LINE                        ;
                                    62    D6 00027           INCL    PAGE_NUMBER                                ; 0706
                                    62    DD 00029           PUSHL   PAGE_NUMBER                                ; 0707
                                    7E    D4 0002B           CLRL    -(SP)                                      ;
                                FC  A2    DD 0002D           PUSHL   REPORT_HEADING_MSG                         ;
                                    7E    D4 00030           CLRL    -(SP)                                      ;
                   7E               01    CE 00032           MNEGL   #1, -(SP)                                  ;
           0000V   CF               05    FB 00035           CALLS   #5, ANL$FORMAT_LINE                        ;
                              FEF4  C2    9F 0003A           PUSHAB  INPUT_FILE_SPEC                            ; 0708
                         00000000G  8F    DD 0003E           PUSHL   #ANLOBJ$_ANYTHING                          ;
                                    7E    D4 00044           CLRL    -(SP)                                      ;
                   7E               01    CE 00046           MNEGL   #1, -(SP)                                  ;
           0000V   CF               04    FB 00049           CALLS   #4, ANL$FORMAT_LINE                        ;
                   7E               01    CE 0004E           MNEGL   #1, -(SP)                                  ; 0709
                   63               01    FB 00051           CALLS   #1, ANL$REPORT_LINE                        ;
                   7E               01    CE 00054           MNEGL   #1, -(SP)                                  ; 0710
                   63               01    FB 00057           CALLS   #1, ANL$REPORT_LINE                        ;
                                       04 0005A 1$:          RET                                                ; 0714

; Routine Size:  91 bytes,    Routine Base:  $CODE$ + 00A0
```

```
1 %sbttl 'ANL$REPORT_LINE - Print a Line in Report'
1 !++
1 ! Functional Description:
1 !       This routine is called to print a line into the report file.
1 !
1 ! Formal Parameters:
1 !       widow_control   Controls widowing as follows:
1 !                               positive        specifies number of lines that
1 !                                               must remain on the page.
1 !                               zero            doesn't matter how many lines.
1 !                               negative        Force line onto current page.
1 !       line            Address of descriptor of line.  Optional.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$report_line(widow_control,line): novalue = begin
2
2 bind
2       line_dsc = .line: descriptor;
2
2 local
2       status: long;
2
2 builtin
2       nullparameter;
2
2
2 ! Don't do anything if we're not generating a report.
2
2 if not .generating_report then
2       return;
2
2 ! If the caller isn't forcing this line onto the page, and there are not
2 ! enough lines left for prevention of widowing, then eject the page.
2
2 if (.widow_control geq 0) and
2    (.line_counter lss .widow_control) then
2       anl$report_page();
2
2 ! Print the line if there is one.  Otherwise put out a blank line.
2
2 if nullparameter(2) then
2       report_rab[rab$w_rsz] = 0
3 else (
3       report_rab[rab$w_rsz] = .line_dsc[len];
;  341        0772  3             report_rab[rab$l_rbf] = .line_dsc[ptr];
;  342        0773  2 );
;  343        0774  2 status = $put(rab=report_rab);
;  344        0775  2 check (.status, anlobj$_writeerr,1,report_spec,.status,.report_rab[rab$l_stv]);
;  345        0776  2
;  346        0777  2 ! Account for the line on the page.
;  347        0778  2
;  348        0779  2 decrement (line_counter);
;  349        0780  2
;  350        0781  2 return;
;  351        0782  2
;  352        0783  1 end;



                                                                   .EXTRN  SYS$PUT

                                                 0004 00000        .ENTRY  ANL$REPORT_LINE, Save R2                        ; 0742
                          52         08    AC    D0 00002          MOVL    LINE, R2                                        ; 0745
                          57      0000'    CF    E9 00006          BLBC    GENERATING_REPORT, 6$                           ; 0756
                                     04    AC    D5 0000B          TSTL    WIDOW_CONTROL                                   ; 0762
                                           0C    19 0000E          BLSS    1$                                              ;
                   04     AC      0000'    CF    D1 00010          CMPL    LINE_COUNTER, WIDOW_CONTROL                     ; 0763
                                           04    18 00016          BGEQ    1$                                              ;
                   89     AF               00    FB 00018          CALLS   #0, ANL$REPORT_PAGE                             ; 0764
                          02               6C    91 0001C 1$:      CMPB    (AP), #2                                        ; 0768
                                           05    1F 0001F          BLSSU   2$                                              ;
                                     08    AC    D5 00021          TSTL    8(AP)                                           ;
                                           06    12 00024          BNEQ    3$                                              ;
                                  0000'    CF    B4 00026 2$:      CLRW    REPORT_RAB+34                                   ; 0769
                                           0B    11 0002A          BRB     4$                                              ;
                0000'     CF               62    B0 0002C 3$:      MOVW    (R2), REPORT_RAB+34                             ; 0771
                0000'     CF         04    A2    D0 00031          MOVL    4(R2), REPORT_RAB+40                            ; 0772
                                  0000'    CF    9F 00037 4$:      PUSHAB  REPORT_RAB                                      ; 0774
            00000000G     00               01    FB 0003B          CALLS   #1, SYS$PUT                                     ;
                          19               50    E8 00042          BLBS    STATUS, 5$                                      ; 0775
                                  0000'    CF    DD 00045          PUSHL   REPORT_RAB+12                                   ;
                                           50    DD 00049          PUSHL   STATUS                                          ;
                                  0000'    CF    9F 0004B          PUSHAB  REPORT_SPEC                                     ;
                                           01    DD 0004F          PUSHL   #1                                              ;
                              00B11004     8F    DD 00051          PUSHL   #11604180                                       ;
            00000000G     00               05    FB 00057          CALLS   #5, LIB$SIGNAL                                  ;
                                  0000'    CF    D7 0005E 5$:      DECL    LINE_COUNTER                                    ; 0779
                                                 04 00062 6$:      RET                                                     ; 0783

; Routine Size:  99 bytes,    Routine Base:  $CODE$ + 00FB
;   354          0784  1 %sbttl 'ANL$FORMAT_LINE - Format Line for Report'
;   355          0785  1 !++
;   356          0786  1 ! Functional Description:
;   357          0787  1 !       This routine is called to format a line and print it in the
;   358          0788  1 !       report file.
;   359          0789  1 !
;   360          0790  1 ! Formal Parameters:
;   361          0791  1 !       widow_control   The number of lines that must be remaining on the
;   362          0792  1 !                       current page.
;   363          0793  1 !       indent_level    The number of tab stops to indent the line.
;   364          0794  1 !       template_msg    The status code of the message defining the line
;   365          0795  1 !                       template.
;   366          0796  1 !       fao1...         $FAO arguments to fill in the template.
;   367          0797  1 !
;   368          0798  1 ! Implicit Inputs:
;   369          0799  1 !       global data
;   370          0800  1 !
;   371          0801  1 ! Implicit Outputs:
;   372          0802  1 !       global data
;   373          0803  1 !
;   374          0804  1 ! Returned Value:
;   375          0805  1 !       none
;   376          0806  1 !
;   377          0807  1 ! Side Effects:
;   378          0808  1 !
;   379          0809  1 !--
;   380          0810  1
;   381          0811  1
;   382          0812  2 global routine anl$format_line(widow_control,indent_level,template_msg,fao1): novalue = begin
;   383          0813  2
;   384          0814  2 local
;   385          0815  2         status: long;
;   386          0816  2
;   387          0817  2
;   388          0818  2 ! First we obtain the text of the template message.
;   389          0819  2
;   390          0820  3 begin
;   391          0821  3 local
;   392          0822  3         local_described_buffer(template_buf,132);
;   393          0823  3
;   394        P 0824  3 status = $getmsg(msgid=.template_msg,
;   395        P 0825  3                  msglen=template_buf,
;   396        P 0826  3                  bufadr=template_buf,
;   397          0827  3                  flags=%b'0001');
;   398          0828  3 check (.status,.status);
;   399          0829  3
;   400          0830  3 ! Now we can plug the $FAO arguments into the message template.
;   401          0831  3
;   402          0832  4 begin
;   403          0833  4 local
;   404          0834  4         local_described_buffer(result_buf,132);
;   405          0835  4
;   406        P 0836  4 status = $faol(ctrstr=template_buf,
;   407        P 0837  4                 outlen=result_buf,
;   408        P 0838  4                 outbuf=result_buf,
;   409          0839  4                 prmlst=fao1);
;   410          0840  4 check (.status,.status);
;  411          0841   4
;  412          0842   4 ! Prefix the resulting text with enough tabs to effect the indentation.
;  413          0843   4
;  414          0844   4 ch$move(.result_buf[len],.result_buf[ptr], .result_buf[ptr]+.indent_level);
;  415          0845   4 result_buf[len] = .result_buf[len] + .indent_level;
;  416          0846   4 ch$fill(%char(tab), .indent_level,.result_buf[ptr]);
;  417          0847   4
;  418          0848   4 ! Print the line, passing along the widow control number.
;  419          0849   4
;  420          0850   4 anl$report_line(.widow_control,result_buf);
;  421          0851   4
;  422          0852   3 end;
;  423          0853   2 end;
;  424          0854   2
;  425          0855   2 return;
;  426          0856   2
;  427          0857   1 end;



                                                                            .EXTRN  SYS$GETMSG, SYS$FAOL

                                                         007C 00000         .ENTRY  ANL$FORMAT_LINE, Save R2,R3,R4,R5,R6         ; 0812
                                   56 00000000G    00    9E 00002           MOVAB   LIB$SIGNAL, R6                               ;
                                   5E     FEE8     CE    9E 00009           MOVAB   -280(SP), SP                                 ;
                        FF74   CD          84      8F    9A 0000E           MOVZBL  #132, TEMPLATE_BUF                           ; 0822
                        FF78   CD        FF7C      CD    9E 00014           MOVAB   TEMPLATE_BUF+8, TEMPLATE_BUF+4               ;
                               7E                  01    7D 0001B           MOVQ    #1, -(SP)                                    ; 0827
                                         FF74      CD    9F 0001E           PUSHAB  TEMPLATE_BUF                                 ;
                                         FF74      CD    9F 00022           PUSHAB  TEMPLATE_BUF                                 ;
                                           0C      AC    DD 00026           PUSHL   TEMPLATE_MSG                                 ;
                    00000000G  00                  05    FB 00029           CALLS   #5, SYS$GETMSG                               ;
                               52                  50    D0 00030           MOVL    R0, STATUS                                   ;
                               05                  52    E8 00033           BLBS    STATUS, 1$                                   ; 0828
                                                   52    DD 00036           PUSHL   STATUS                                       ;
                               66                  01    FB 00038           CALLS   #1, LIB$SIGNAL                               ;
                               6E          84      8F    9A 0003B   1$:     MOVZBL  #132, RESULT_BUF                             ; 0834
                        04     AE          08      AE    9E 0003F           MOVAB   RESULT_BUF+8, RESULT_BUF+4                   ;
                                           10      AC    9F 00044           PUSHAB  FAO1                                         ; 0839
                                           04      AE    9F 00047           PUSHAB  RESULT_BUF                                   ;
                                           08      AE    9F 0004A           PUSHAB  RESULT_BUF                                   ;
                                         FF74      CD    9F 0004D           PUSHAB  TEMPLATE_BUF                                 ;
                    00000000G  00                  04    FB 00051           CALLS   #4, SYS$FAOL                                 ;
                               52                  50    D0 00058           MOVL    R0, STATUS                                   ;
                               05                  52    E8 0005B           BLBS    STATUS, 2$                                   ; 0840
                                                   52    DD 0005E           PUSHL   STATUS                                       ;
                               66                  01    FB 00060           CALLS   #1, LIB$SIGNAL                               ;
                  50    04     AE          08      AC    C1 00063   2$:     ADDL3   INDENT_LEVEL, RESULT_BUF+4, R0               ; 0844
                  60    04     BE                  6E    28 00069           MOVC3   RESULT_BUF, @RESULT_BUF+4, (R0)              ;
                               6E          08      AC    A0 0006E           ADDW2   INDENT_LEVEL, RESULT_BUF                     ; 0845
      08    AC    09           6E                  00    2C 00072           MOVC5   #0, (SP), #9, INDENT_LEVEL, @RESULT_BUF+4    ; 0846
                                           04      BE       00078                                                                ;
                                                   5E    DD 0007A           PUSHL   SP                                           ; 0850
                                           04      AC    DD 0007C           PUSHL   WIDOW_CONTROL                                ;
                        FF19   CF                  02    FB 0007F           CALLS   #2, ANL$REPORT_LINE                          ;
                                                      04 00084              RET                                                  ; 0857
```

```
; Routine Size:  133 bytes,    Routine Base:  $CODE$ + 015E
```

```
1 %sbttl 'ANL$FORMAT_ERROR - Put Error Message in Report'
1 !++
1 ! Functional Description:
1 !       This routine is called to format an error message into the report
1 !       file.
1 !
1 ! Formal Parameters:
1 !       error_msg       Status code for the error message.
1 !       fao1...         $FAO substitution parameters for the message.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$format_error(error_msg,fao1,fao2,fao3,fao4): novalue = begin
2
2 bind
2       flag_string = describe('***  ');
2
2 builtin
2       actualcount;
2
2
2 ! We case on the number of $FAO parameters and call ANL$FORMAT_LINE to
2 ! do the work.  In all cases, however, we add our own first parameter,
2 ! which is the error message flag string.
2
2 case actualcount() from 1 to 5 of set
2 [1]:    anl$format_line(-1,0,.error_msg,flag_string);
2 [2]:    anl$format_line(-1,0,.error_msg,flag_string,.fao1);
2 [3]:    anl$format_line(-1,0,.error_msg,flag_string,.fao1,.fao2);
2 [4]:    anl$format_line(-1,0,.error_msg,flag_string,.fao1,.fao2,.fao3);
2 [5]:    anl$format_line(-1,0,.error_msg,flag_string,.fao1,.fao2,.fao3,.fao4);
2 tes;
2
2 ! Keep track of the number of errors reported.  Also keep track of
2 ! most severe error which has occurred.
2
2 increment (error_count);
2 if   severity_level (.error_msg) gtr
3    severity_level (.anl$worst_error)          ! If higher than watermark
2 then anl$worst_error = .error_msg;            !  -then set new worst error
2
2 return;
2
1 end;
```

```
                                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

                       20  20  2A  2A  2A  00028 P.AAG:       .ASCII  \***  \                                    ;
                                           0002D              .BLKB   3
                               00000005    00030 P.AAF:       .LONG   5                                          ;
                               00000000'   00034              .ADDRESS P.AAG                                     ;

                                                 FLAG_STRING=         P.AAF

                                                              .PSECT  $CODE$,NOWRT,2

                                      003C 00000              .ENTRY  ANL$FORMAT_ERROR, Save R2,R3,R4,R5          ; 0882
                       55     0000'   CF  9E 00002            MOVAB   FLAG_STRING, R5                            ;
                       54     FF70    CF  9E 00007            MOVAB   ANL$FORMAT_LINE, R4                        ;
                       52       04    AC  D0 0000C            MOVL    ERROR_MSG, R2                              ; 0896
               04      01             6C  8F 00010            CASEB   (AP), #1, #4                               ; 0895
 0035        0025    0016           000A     00014 1$:        .WORD   2$-1$,-                                    ;
                                    0048     0001C                    3$-1$,-                                    ;
                                                                      4$-1$,-                                    ;
                                                                      5$-1$,-                                    ;
                                                                      6$-1$                                      ;
                                      24  BB 0001E 2$:        PUSHR   #^M<R2,R5>                                 ; 0896
                                      7E  D4 00020            CLRL    -(SP)                                      ;
                       7E             01  CE 00022            MNEGL   #1, -(SP)                                  ;
                       64             04  FB 00025            CALLS   #4, ANL$FORMAT_LINE                        ;
                                      44  11 00028            BRB     7$                                         ;
                                08    AC  DD 0002A 3$:        PUSHL   FA01                                       ; 0897
                                      24  BB 0002D            PUSHR   #^M<R2,R5>                                 ;
                                      7E  D4 0002F            CLRL    -(SP)                                      ;
                       7E             01  CE 00031            MNEGL   #1, -(SP)                                  ;
                       64             05  FB 00034            CALLS   #5, ANL$FORMAT_LINE                        ;
                                      35  11 00037            BRB     7$                                         ;
                       7E       08    AC  7D 00039 4$:        MOVQ    FA01, -(SP)                                ; 0898
                                      24  BB 0003D            PUSHR   #^M<R2,R5>                                 ;
                                      7E  D4 0003F            CLRL    -(SP)                                      ;
                       7E             01  CE 00041            MNEGL   #1, -(SP)                                  ;
                       64             06  FB 00044            CALLS   #6, ANL$FORMAT_LINE                        ;
                                      25  11 00047            BRB     7$                                         ;
                       7E       0C    AC  7D 00049 5$:        MOVQ    FA02, -(SP)                                ; 0899
                                08    AC  DD 0004D            PUSHL   FA01                                       ;
                                      24  BB 00050            PUSHR   #^M<R2,R5>                                 ;
                                      7E  D4 00052            CLRL    -(SP)                                      ;
                       7E             01  CE 00054            MNEGL   #1, -(SP)                                  ;
                       64             07  FB 00057            CALLS   #7, ANL$FORMAT_LINE                        ;
                                      12  11 0005A            BRB     7$                                         ;
                       7E       10    AC  7D 0005C 6$:        MOVQ    FA03, -(SP)                                ; 0900
                       7E       08    AC  7D 00060            MOVQ    FA01, -(SP)                                ;
                                      24  BB 00064            PUSHR   #^M<R2,R5>                                 ;
                                      7E  D4 00066            CLRL    -(SP)                                      ;
                       7E             01  CE 00068            MNEGL   #1, -(SP)                                  ;
                       64             08  FB 0006B            CALLS   #8, ANL$FORMAT_LINE                        ;
                              0000'   CF  D6 0006E 7$:        INCL    ERROR_COUNT                                ; 0906
                       50             52  D0 00072            MOVL    R2, TMP_CODE                               ; 0907
   51          50      03             00  EF 00075            EXTZV   #0, #3, TMP_CODE, R1                       ;
```

```
        50         50            01            00  EF 0007A           EXTZV   #0, #1, TMP_CODE, R0                          ;
                                 50            04  C4 0007F           MULL2   #4, R0                                        ;
                                 51            50  C2 00082           SUBL2   R0, R1                                        ;
                                 51            03  C0 00085           ADDL2   #3, R1                                        ;
                                 50     0000'  CF  D0 00088           MOVL    ANL$WORST_ERROR, TMP_CODE                     ; 0908
        53         50            03            00  EF 0008D           EXTZV   #0, #3, TMP_CODE, R3                          ;
        50         50            01            00  EF 00092           EXTZV   #0, #1, TMP_CODE, R0                          ;
                                 50            04  C4 00097           MULL2   #4, R0                                        ;
                                 53            50  C2 0009A           SUBL2   R0, R3                                        ;
                                 50       03   A3  9E 0009D           MOVAB   3(R3), R0                                     ;
                                 50            51  D1 000A1           CMPL    R1, R0                                        ;
                                               05  15 000A4           BLEQ    8$                                            ;
                          0000'  CF            52  D0 000A6           MOVL    R2, ANL$WORST_ERROR                           ; 0909
                                                   04 000AB 8$:       RET                                                   ; 0913

; Routine Size:  172 bytes,    Routine Base:  $CODE$ + 01E3
;  486          0914   1 %sbttl 'ANL$ERROR_COUNT - Report Count of Errors'
;  487          0915   1 !++
;  488          0916   1 ! Functional Description:
;  489          0917   1 !       This routine is called to print a line telling how many errors
;  490          0918   1 !       were discovered during the analysis.
;  491          0919   1 !
;  492          0920   1 ! Formal Parameters:
;  493          0921   1 !       none
;  494          0922   1 !
;  495          0923   1 ! Implicit Inputs:
;  496          0924   1 !       global data
;  497          0925   1 !
;  498          0926   1 ! Implicit Outputs:
;  499          0927   1 !       global data
;  500          0928   1 !
;  501          0929   1 ! Returned Value:
;  502          0930   1 !       none
;  503          0931   1 !
;  504          0932   1 ! Side Effects:
;  505          0933   1 !
;  506          0934   1 !--
;  507          0935   1
;  508          0936   1
;  509          0937   2 global routine anl$error_count: novalue = begin
;  510          0938   2
;  511          0939   2
;  512          0940   2 ! First we print the error count in the report.
;  513          0941   2
;  514          0942   2 if .error_count eqlu 0 then
;  515          0943   2       anl$format_line(0,0,anlobj$_errornone)
;  516          0944   2 else
;  517          0945   2       anl$format_line(0,0,anlobj$_errorcount,.error_count);
;  518          0946   2 anl$report_line(0);
;  519          0947   2 anl$report_line(0);
;  520          0948   2
;  521          0949   2 ! If the report is not going to SYS$OUTPUT, we also want to display one line
;  522          0950   2 ! for the user at the terminal.  This contains the report heading text and
;  523          0951   2 ! the error count.
;  524          0952   2
;  525          0953   2 if ch$neq(.report_spec[len],.report_spec[ptr], 10,uplit byte('SYS$OUTPUT'),' ') then
;  526          0954   2       signal(anlobj$_errors,2,input_file_spec,.error_count);
;  527          0955   2
;  528          0956   2 ! Now we can reset the error counter for the next file.
;  529          0957   2
;  530          0958   2 error_count = 0;
;  531          0959   2
;  532          0960   2 return;
;  533          0961   2
;  534          0962   1 end;


                                                             .PSECT  $PLIT$,NOWRT,NOEXE,2

         54  55  50  54  55  4F  24  53  59  53  00038 P.AAH:  .ASCII  \SYS$OUTPUT\                                    ;
```

```
                                                              .PSECT  $CODE$,NOWRT,2

                                          003C 00000         .ENTRY  ANL$ERROR_COUNT, Save R2,R3,R4,R5                ; 0937
                      55      FEC9   CF   9E 00002           MOVAB   ANL$FORMAT_LINE, R5                              ;
                      54      0000'  CF   9E 00007           MOVAB   ERROR_COUNT, R4                                  ;
                      50             64   D0 0000C           MOVL    ERROR_COUNT, R0                                  ; 0942
                                     0D   12 0000F           BNEQ    1$                                               ;
                        00000000G    8F   DD 00011           PUSHL   #ANLOBJ$_ERRORNONE                               ; 0943
                                     7E   7C 00017           CLRQ    -(SP)                                            ;
                      65             03   FB 00019           CALLS   #3, ANL$FORMAT_LINE                              ;
                                     0D   11 0001C           BRB     2$                                               ;
                                     50   DD 0001E  1$:      PUSHL   R0                                               ; 0945
                        00000000G    8F   DD 00020           PUSHL   #ANLOBJ$_ERRORCOUNT                              ;
                                     7E   7C 00026           CLRQ    -(SP)                                            ;
                      65             04   FB 00028           CALLS   #4, ANL$FORMAT_LINE                              ;
                                     7E   D4 0002B  2$:      CLRL    -(SP)                                            ; 0946
                 9D   A5             01   FB 0002D           CALLS   #1, ANL$REPORT_LINE                              ;
                                     7E   D4 00031           CLRL    -(SP)                                            ; 0947
                 9D   A5             01   FB 00033           CALLS   #1, ANL$REPORT_LINE                              ;
 0A        20   FD50  D4      FD4C   C4   2D 00037           CMPC5   REPORT_SPEC, @REPORT_SPEC+4, #32, #10, -         ; 0953
                              0000'  CF      00040                   P.AAH                                            ;
                                     15   13 00043           BEQL    3$                                               ;
                                     64   DD 00045           PUSHL   ERROR_COUNT                                      ; 0954
                              FEEC   C4   9F 00047           PUSHAB  INPUT_FILE_SPEC                                  ;
                                     02   DD 0004B           PUSHL   #2                                               ;
                        00000000G    8F   DD 0004D           PUSHL   #ANLOBJ$_ERRORS                                  ;
           00000000G  00             04   FB 00053           CALLS   #4, LIB$SIGNAL                                   ;
                                     64   D4 0005A  3$:      CLRL    ERROR_COUNT                                      ; 0958
                                          04 0005C           RET                                                      ; 0962

; Routine Size:  93 bytes,    Routine Base:  $CODE$ + 028F
;   536         0963   1 %sbttl 'ANL$EXIT_WITH_STATUS - Exit to VMS With a Status'
;   537         0964   1 !++
;   538         0965   1 ! Functional Description:
;   539         0966   1 !       This routine is called when it's time to exit back to VMS.  We
;   540         0967   1 !       exit with the status in anl$worst_error.  (This contains
;   541         0968   1 !       success status if no errors have occurred.)
;   542         0969   1 !
;   543         0970   1 ! Formal Parameters:
;   544         0971   1 !       none
;   545         0972   1 !
;   546         0973   1 ! Implicit Inputs:
;   547         0974   1 !       global data
;   548         0975   1 !
;   549         0976   1 ! Implicit Outputs:
;   550         0977   1 !       global data
;   551         0978   1 !
;   552         0979   1 ! Returned Value:
;   553         0980   1 !       does not return
;   554         0981   1 !
;   555         0982   1 ! Side Effects:
;   556         0983   1 !
;   557         0984   1 !--
;   558         0985   1
;   559         0986   1
;   560         0987   2 global routine anl$exit_with_status: novalue = begin
;   561         0988   2
;   562         0989   2 ! if it was an interactive session, always return success.  otherwise
;   563         0990   2 ! return worst error
;   564         0991   2
;   565         0992   2 if .anl$gb_interactive then
;   566         0993   3       $exit(code=anlobj$_ok)
;   567         0994   2 else
;   568         0995   2       $exit(code=.anl$worst_error or sts$m_inhib_msg);
;   569         0996   2
;   570         0997   1 end;


                                                                        .EXTRN  SYS$EXIT

                                                  0000 00000            .ENTRY  ANL$EXIT_WITH_STATUS, Save nothing              ; 0987
                          08       0000G   CF       E9 00002            BLBC    ANL$GB_INTERACTIVE, 1$                          ; 0992
                               00000000G   8F       DD 00007            PUSHL   #ANLOBJ$_OK                                     ; 0993
                                           0A       11 0000D            BRB     2$                                              ;
                   7E      0000'   CF 10000000   8F   C9 0000F 1$:       BISL3   #268435456, ANL$WORST_ERROR, -(SP)              ; 0995
                      00000000G   00           01   FB 00019 2$:       CALLS   #1, SYS$EXIT                                    ;
                                                    04 00020            RET                                                     ; 0997

; Routine Size:  33 bytes,    Routine Base:  $CODE$ + 02EC
```

```
1 %sbttl 'ANL$FORMAT_HEX - Format Hex Dump of Data'
1 !++
1 ! Functional Description:
1 !       This routine is called to format a hex dump of some bytes.
1 !       It includes the character representation of the bytes also.
1 !
1 ! Formal Parameters:
1 !       indent_level    The indentation level at which to place the dump.
1 !       data            Address of descriptor of data to be dumped.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$format_hex(indent_level,data): novalue = begin
2
2 bind
2       data_dsc = .data: descriptor,
2       data_vector = .data_dsc[ptr]: vector[,byte];
2
2 local
2       i: long,
2       arg_list: vector[20,long],
2       count: long;
2
2 builtin
2       callg;
2
2
2 ! If the data is null, just quit.
2
2 if .data_dsc[len] eqlu 0 then
2       return;
2
2 ! We begin by printing two heading lines.  The first shows the offsets
2 ! of the bytes and the second is a line of dashes.
2
2 anl$format_line(3,.indent_level,anlobj$_hexheading1);
2 anl$format_line(0,.indent_level,anlobj$_hexheading2);
2
2 ! We will be builing argument lists to ANL$FORMAT_LINE.  It will always
2 ! include widow control, indentation level, and the message code.
2
2 arg_list[1] = 0;
2 arg_list[2] = .indent_level;
2 arg_list[3] = anlobj$_hexdata;
2
```

```
;  629          1055  2 ! Now we go into a loop, once through for each 8 bytes to be formatted.
;  630          1056  2
;  631          1057  2 i = 0;
;  632          1058  3 while .i lssu .data_dsc[len] do (
;  633          1059  3
;  634          1060  3         ! Calculate the number of bytes that will go on this line.
;  635          1061  3
;  636          1062  3         count = minu(.data_dsc[len]-.i,8);
;  637          1063  3
;  638          1064  3         ! Next in the argument list we need a count of the spaces to skip
;  639          1065  3         ! so the bytes will be lined up from right to left.
;  640          1066  3
;  641          1067  3         arg_list[4] = (8 - .count) * 3;
;  642          1068  3
;  643          1069  3         ! Now we need the count itself.
;  644          1070  3
;  645          1071  3         arg_list[5] = .count;
;  646          1072  3
;  647          1073  3         ! Now we loop through 8 (or less) bytes and put them in the
;  648          1074  3         ! argument list (backwards, of course).
;  649          1075  3
;  650          1076  4         decr j from .count-1 to 0 do (
;  651          1077  4                 arg_list[6+.j] = .data_vector[.i];
;  652          1078  4                 increment (i);
;  653          1079  3         );
;  654          1080  3
;  655          1081  3         ! Next we have the byte offset.
;  656          1082  3
;  657          1083  3         arg_list[6+.count] = .i - .count;
;  658          1084  3
;  659          1085  3         ! Now we have to add to the argument list the byte count and a
;  660          1086  3         ! pointer to the byte string.
;  661          1087  3
;  662          1088  3         arg_list[7+.count] = .count;
;  663          1089  3         arg_list[8+.count] = data_vector[.i - .count];
;  664          1090  3
;  665          1091  3         ! Finally, fill in the argument count.
;  666          1092  3
;  667          1093  3         arg_list[0] = 8 + .count;
;  668          1094  3
;  669          1095  3         ! Now we can print the hex data.
;  670          1096  3
;  671          1097  3         callg(arg_list,anl$format_line);
;  672          1098  2 );
;  673          1099  2
;  674          1100  2 return;
;  675          1101  2
;  676          1102  1 end;



                                    003C 00000        .ENTRY  ANL$FORMAT_HEX, Save R2,R3,R4,R5              ; 1022
                   55      FE4B   CF   9E 00002        MOVAB   ANL$FORMAT_LINE, R5                           ;
                   5E        B0   AE   9E 00007        MOVAB   -80(SP), SP                                   ;
                   54        08   AC   D0 0000B        MOVL    DATA, R4                                      ; 1025
```

```
                                                 64   B5 0000F        TSTW     (R4)                                          ; 1039
                                                 33   13 00011        BEQL     2$                                            ;
                                        00000000G  8F   DD 00013        PUSHL    #ANLOBJ$_HEXHEADING1                          ; 1045
                                               04   AC   DD 00019        PUSHL    INDENT_LEVEL                                  ;
                                                 03   DD 0001C        PUSHL    #3                                            ;
                                     65          03   FB 0001E        CALLS    #3, ANL$FORMAT_LINE                           ;
                                        00000000G  8F   DD 00021        PUSHL    #ANLOBJ$_HEXHEADING2                          ; 1046
                                               04   AC   DD 00027        PUSHL    INDENT_LEVEL                                  ;
                                                 7E   D4 0002A        CLRL     -(SP)                                         ;
                                     65          03   FB 0002C        CALLS    #3, ANL$FORMAT_LINE                           ;
                                               04   AE   D4 0002F        CLRL     ARG_LIST+4                                    ; 1051
                              08     AE        04   AC   D0 00032        MOVL     INDENT_LEVEL, ARG_LIST+8                      ; 1052
                              0C     AE 00000000G  8F   D0 00037        MOVL     #ANLOBJ$_HEXDATA, ARG_LIST+12                 ; 1053
                                                 53   D4 0003F        CLRL     I                                             ; 1057
          53           64            10          00   ED 00041 1$:    CMPZV    #0, #16, (R4), I                              ; 1058
                                                 4F   1B 00046 2$:    BLEQU    6$                                            ;
                                     50          64   3C 00048        MOVZWL   (R4), R0                                      ; 1062
                                     50          53   C2 0004B        SUBL2    I, R0                                         ;
                                     08          50   D1 0004E        CMPL     R0, #8                                        ;
                                                 03   1B 00051        BLEQU    3$                                            ;
                                     50          08   D0 00053        MOVL     #8, R0                                        ;
                                     52          50   D0 00056 3$:    MOVL     R0, COUNT                                     ;
                                     50        F8   A2   9E 00059        MOVAB    -8(R2), R0                                    ; 1067
                                     50          03   C4 0005D        MULL2    #3, R0                                        ;
                              10     AE          50   CE 00060        MNEGL    R0, ARG_LIST+16                               ;
                              14     AE          52   D0 00064        MOVL     COUNT, ARG_LIST+20                            ; 1071
                                     50          52   D0 00068        MOVL     COUNT, J                                      ; 1076
                                                 09   11 0006B        BRB      5$                                            ;
                              18 AE40        04 B443   9A 0006D 4$:    MOVZBL   @4(R4)[I], ARG_LIST+24[J]                     ; 1077
                                                 53   D6 00074        INCL     I                                             ; 1078
                                     F4          50   F4 00076 5$:    SOBGEQ   J, 4$                                         ; 1076
                       50            53          52   C3 00079        SUBL3    COUNT, I, R0                                  ; 1083
                              18 AE42            50   D0 0007D        MOVL     R0, ARG_LIST+24[COUNT]                        ;
                              1C AE42            52   D0 00082        MOVL     COUNT, ARG_LIST+28[COUNT]                     ; 1088
                              20 AE42        04 B440   9E 00087        MOVAB    @4(R4)[R0], ARG_LIST+32[COUNT]                ; 1089
                                     6E        08   A2   9E 0008E        MOVAB    8(R2), ARG_LIST                               ; 1093
                                     65          6E   FA 00092        CALLG    ARG_LIST, ANL$FORMAT_LINE                     ; 1097
                                                 AA   11 00095        BRB      1$                                            ; 1058
                                                      04 00097 6$:    RET                                                    ; 1102

; Routine Size:  152 bytes,    Routine Base:  $CODE$ + 030D
```

```
1 %sbttl 'ANL$FORMAT_FLAGS - Format Flag Bits'
1 !++
1 ! Functional Description:
1 !       This routine is called to format the flags in a byte/word/longword
1 !       of flags.
1 !
1 ! Formal Parameters:
1 !       indent_level    The level at which the introductory message is to
1 !                       be indented.  The flags are indented one more level.
1 !       intro_msg       The introductory message.
1 !       flags           The flag bits.
1 !       flag_def        A longword vector defining the flags.  The zeroth
1 !                       entry specifies the highest-numbered flag.  The
1 !                       remaining longwords contain the address of a counted
1 !                       string giving the name of the flag.  If the flag is
1 !                       undefined, the longword contains zero.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$format_flags(indent_level,intro_msg,flags,flag_def): novalue = begin
2
2 bind
2       flags_vector = flags: bitvector[],
2       flag_def_vector = .flag_def: vector[,long];
2
2 local
2       i: long;
2
2
2 ! Begin by printing the introductory message.
2
2 anl$format_line(2,.indent_level,.intro_msg);
2
2 ! Now we loop through the flags and process each one that is defined.
2 ! We print the flag name, bit number, and current setting.
2
3 incru i from 0 to .flag_def_vector[0] do (
3       if .flag_def_vector[.i+1] nequ 0 then
3               anl$format_line(0,.indent_level+1,anlobj$_flag,
3                               .i,.flag_def_vector[.i+1],.flags_vector[.i]);
2 );
2
2 return;
2
1 end;
```

```
                                               0004 00000          .ENTRY  ANL$FORMAT_FLAGS, Save R2                        : 1134
                          7E         04     AC   7D 00002          MOVQ    INDENT_LEVEL, -(SP)                              : 1146
                                            02   DD 00006          PUSHL   #2                                               :
                  FDAC    CF                03   FB 00008          CALLS   #3, ANL$FORMAT_LINE                              :
                                            52   D4 0000D          CLRL    I                                                : 1151
                                            29   11 0000F          BRB     3$                                               :
                          50         10 BC42     DE 00011 1$:      MOVAL   @FLAG_DEF[I], R0                                 : 1152
                                     04     A0   D5 00016          TSTL    4(R0)                                            :
                                            1D   13 00019          BEQL    2$                                               :
      7E       0C    AC                01   52   EF 0001B          EXTZV   I, #1, FLAGS_VECTOR, -(SP)                       : 1154
                                     04     A0   DD 00021          PUSHL   4(R0)                                            :
                                            52   DD 00024          PUSHL   I                                                :
                               000000006    8F   DD 00026          PUSHL   #ANLOBJ$_FLAG                                    : 1153
                     7E      04    AC       01   C1 0002C          ADDL3   #1, INDENT_LEVEL, -(SP)                          :
                                            7E   D4 00031          CLRL    -(SP)                                            :
                  FD81    CF                06   FB 00033          CALLS   #6, ANL$FORMAT_LINE                              :
                                            52   D6 00038 2$:      INCL    I                                                : 1151
                    10    BC                52   D1 0003A 3$:      CMPL    I, @FLAG_DEF                                     :
                                            D1   1B 0003E          BLEQU   1$                                               :
                                                 04 00040          RET                                                      : 1159

; Routine Size:  65 bytes,    Routine Base:  $CODE$ + 03A5
```

```
%sbttl 'ANL$FORMAT_DATA_TYPE - Format a Data Type'
!++
! Functional Description:
!       This routine is called to format a nice line for a data type,
!       as defined in the VAX architecture manual.
!
! Formal Parameters:
!       indent_level    The level of indentation for the line.
!       data_type       The data type byte.
!
! Implicit Inputs:
!       global data
!
! Implicit Outputs:
!       global data
!
! Returned Value:
!       none
!
! Side Effects:
!
!--


global routine anl$format_data_type(indent_level,data_type): novalue = begin


own
        data_type_table: vector[33,long] initial(
                                        uplit byte(%ascic 'Z'),
                                        uplit byte(%ascic 'V'),
                                        uplit byte(%ascic 'BU'),
                                        uplit byte(%ascic 'WU'),
                                        uplit byte(%ascic 'LU'),
                                        uplit byte(%ascic 'QU'),
                                        uplit byte(%ascic 'B'),
                                        uplit byte(%ascic 'W'),
                                        uplit byte(%ascic 'L'),
                                        uplit byte(%ascic 'Q'),
                                        uplit byte(%ascic 'F'),
                                        uplit byte(%ascic 'D'),
                                        uplit byte(%ascic 'FC'),
                                        uplit byte(%ascic 'DC'),
                                        uplit byte(%ascic 'T'),
                                        uplit byte(%ascic 'NU'),
                                        uplit byte(%ascic 'NL'),
                                        uplit byte(%ascic 'NLO'),
                                        uplit byte(%ascic 'NR'),
                                        uplit byte(%ascic 'NRO'),
                                        uplit byte(%ascic 'NZ'),
                                        uplit byte(%ascic 'P'),
                                        uplit byte(%ascic 'ZI'),
                                        uplit byte(%ascic 'ZEM'),
                                        uplit byte(%ascic 'DSC'),
                                        uplit byte(%ascic 'OU'),
                                        uplit byte(%ascic 'O'),
                                        uplit byte(%ascic 'G'),
          2                                    uplit byte(%ascic 'H'),
          2                                    uplit byte(%ascic 'GC'),
          2                                    uplit byte(%ascic 'HC'),
          2                                    uplit byte(%ascic 'CIT'),
          2                                    uplit byte(%ascic 'BPV'));
          2
          2
          2 ! If it is a standard data type, print it's name and number.  Otherwise just
          2 ! use the number.
          2
          2 anl$format_line(0,.indent_level,anlobj$_datatype,
          3                (if .data_type lssu %allocation(data_type_table)/4 then .data_type_table[.data_type]
          2                                                                   else uplit byte(%ascic '???')),
          2                .data_type);
          2
          2 return;
          2
          1 end;


                                                          .PSECT  $PLIT$,NOWRT,NOEXE,2

                        5A  01  00042 P.AAI:   .ASCII  <1>\Z\
                        56  01  00044 P.AAJ:   .ASCII  <1>\V\
                    55  42  02  00046 P.AAK:   .ASCII  <2>\BU\
                    55  57  02  00049 P.AAL:   .ASCII  <2>\WU\
                    55  4C  02  0004C P.AAM:   .ASCII  <2>\LU\
                    55  51  02  0004F P.AAN:   .ASCII  <2>\QU\
                        42  01  00052 P.AAO:   .ASCII  <1>\B\
                        57  01  00054 P.AAP:   .ASCII  <1>\W\
                        4C  01  00056 P.AAQ:   .ASCII  <1>\L\
                        51  01  00058 P.AAR:   .ASCII  <1>\Q\
                        46  01  0005A P.AAS:   .ASCII  <1>\F\
                        44  01  0005C P.AAT:   .ASCII  <1>\D\
                    43  46  02  0005E P.AAU:   .ASCII  <2>\FC\
                    43  44  02  00061 P.AAV:   .ASCII  <2>\DC\
                        54  01  00064 P.AAW:   .ASCII  <1>\T\
                    55  4E  02  00066 P.AAX:   .ASCII  <2>\NU\
                    4C  4E  02  00069 P.AAY:   .ASCII  <2>\NL\
                4F  4C  4E  03  0006C P.AAZ:   .ASCII  <3>\NLO\
                    52  4E  02  00070 P.ABA:   .ASCII  <2>\NR\
                4F  52  4E  03  00073 P.ABB:   .ASCII  <3>\NRO\
                    5A  4E  02  00077 P.ABC:   .ASCII  <2>\NZ\
                        50  01  0007A P.ABD:   .ASCII  <1>\P\
                    49  5A  02  0007C P.ABE:   .ASCII  <2>\ZI\
                4D  45  5A  03  0007F P.ABF:   .ASCII  <3>\ZEM\
                43  53  44  03  00083 P.ABG:   .ASCII  <3>\DSC\
                    55  4F  02  00087 P.ABH:   .ASCII  <2>\OU\
                        4F  01  0008A P.ABI:   .ASCII  <1>\O\
                        47  01  0008C P.ABJ:   .ASCII  <1>\G\
                        48  01  0008E P.ABK:   .ASCII  <1>\H\
                    43  47  02  00090 P.ABL:   .ASCII  <2>\GC\
                    43  48  02  00093 P.ABM:   .ASCII  <2>\HC\
                54  49  43  03  00096 P.ABN:   .ASCII  <3>\CIT\
                56  50  42  03  0009A P.ABO:   .ASCII  <3>\BPV\
                3F  3F  3F  03  0009E P.ABP:   .ASCII  <3>\???\
                                                                              .PSECT  $OWN$,NOEXE,2

00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 002B8 DATA_TYPE_TABLE:
                                                                              .ADDRESS P.AAI, P.AAJ, P.AAK, P.AAL, P.AAM, -          ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 002D0                      P.AAN, P.AAO, P.AAP, P.AAQ, P.AAR, P.AAS, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 002E8                      P.AAT, P.AAU, P.AAV, P.AAW, P.AAX, P.AAY, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00300                      P.AAZ, P.ABA, P.ABB, P.ABC, P.ABD, P.ABE, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00318                      P.ABF, P.ABG, P.ABH, P.ABI, P.ABJ, P.ABK, -   ;
                              00000000' 00000000' 00000000' 00330                      P.ABL, P.ABM, P.ABN, P.ABO                    ;



                                                                              .PSECT  $CODE$,NOWRT,2

                                                   0000 00000                 .ENTRY  ANL$FORMAT_DATA_TYPE, Save nothing            ; 1184
                                        50      08   AC   D0 00002            MOVL    DATA_TYPE, R0                                 ; 1230
                                                     50   DD 00006            PUSHL   R0                                            ;
                                        21           50   D1 00008            CMPL    R0, #33                                       ; 1228
                                                     07   1E 0000B            BGEQU   1$                                            ;
                                             0000'CF40    DD 0000D            PUSHL   DATA_TYPE_TABLE[R0]                           ;
                                                     07   11 00012            BRB     2$                                            ;
                                        50    0000'  CF   9E 00014 1$:        MOVAB   P.ABP, R0                                     ; 1229
                                                     50   DD 00019            PUSHL   R0                                            ;
                                          00000000G  8F   DD 0001B 2$:        PUSHL   #ANLOBJ$_DATATYPE                             ; 1227
                                                04   AC   DD 00021            PUSHL   INDENT_LEVEL                                  ;
                                                     7E   D4 00024            CLRL    -(SP)                                         ;
                                 FD4D   CF           05   FB 00026            CALLS   #5, ANL$FORMAT_LINE                           ;
                                                        04 0002B              RET                                                   ; 1234

; Routine Size:  44 bytes,    Routine Base:  $CODE$ + 03E6
```

```
%sbttl 'ANL$FORMAT_MASK - Format an Entry Mask'
!++
! Functional Description:
!       This routine is called to format an entry mask word.
!
! Formal Parameters:
!       indent_level    The level of indentation for the mask.
!       mask            The mask itself.
!
! Implicit Inputs:
!       global data
!
! Implicit Outputs:
!       global data
!
! Returned Value:
!       none
!
! Side Effects:
!
!--


global routine anl$format_mask(indent_level,mask): novalue = begin

bind
        mask_vector = mask: bitvector[16];

own
        bit_name: vector[16,long] initial(
                        'R0,',  'R1,',  'R2,',  'R3,',
                        'R4,',  'R5,',  'R6,',  'R7,',
                        'R8,',  'R9,',  'R10,', 'R11,',
                        '--,',  '--,',  'IV,',  'DV,');

local
        i: long,
        bit_name_len: long;
local
        local_described_buffer(mask_buf,64);

! We are going to scan the entry mask and concatenate together the names
! of the bits that are on.

mask_buf[len] = 0;
incru i from 0 to 15 do (
        if .mask_vector[.i] then (
                bit_name_len = (if .i eqlu 10 or .i eqlu 11 then 4 else 3);
                ch$move(.bit_name_len,bit_name[.i], .mask_buf[ptr]+.mask_buf[len]);
                mask_buf[len] = .mask_buf[len] + .bit_name_len;
        );
);

! If any bits were set in the mask, we will have a spurious trailing comma.
! Get rid of it.
```

```
;  869          1292  2 if .mask_buf[len] gtru 0 then
;  870          1293  2         decrement (mask_buf[len]);
;  871          1294  2
;  872          1295  2 ! Now we can print the mask.
;  873          1296  2
;  874          1297  2 anl$format_line(0,.indent_level,anlobj$_mask,mask_buf);
;  875          1298  2
;  876          1299  2 return;
;  877          1300  2
;  878          1301  1 end;



                                                                .PSECT  $OWN$,NOEXE,2

                               00  2C  30  52  0033C BIT_NAME:
                                                                .ASCII  \R0,\<0>                                  ;
                               00  2C  31  52  00340            .ASCII  \R1,\<0>                                  ;
                               00  2C  32  52  00344            .ASCII  \R2,\<0>                                  ;
                               00  2C  33  52  00348            .ASCII  \R3,\<0>                                  ;
                               00  2C  34  52  0034C            .ASCII  \R4,\<0>                                  ;
                               00  2C  35  52  00350            .ASCII  \R5,\<0>                                  ;
                               00  2C  36  52  00354            .ASCII  \R6,\<0>                                  ;
                               00  2C  37  52  00358            .ASCII  \R7,\<0>                                  ;
                               00  2C  38  52  0035C            .ASCII  \R8,\<0>                                  ;
                               00  2C  39  52  00360            .ASCII  \R9,\<0>                                  ;
                               2C  30  31  52  00364            .ASCII  \R10,\                                    ;
                               2C  31  31  52  00368            .ASCII  \R11,\                                    ;
                               00  2C  2D  2D  0036C            .ASCII  \--,\<0>                                  ;
                               00  2C  2D  2D  00370            .ASCII  \--,\<0>                                  ;
                               00  2C  56  49  00374            .ASCII  \IV,\<0>                                  ;
                               00  2C  56  44  00378            .ASCII  \DV,\<0>                                  ;



                                                                .PSECT  $CODE$,NOWRT,2

                                        00FC 00000              .ENTRY  ANL$FORMAT_MASK, Save R2,R3,R4,R5,R6,R7   ; 1258
                    5E        BC    AE    9E 00002              MOVAB   -68(SP), SP                               ;
                    7E        40    8F    9A 00006              MOVZBL  #64, MASK_BUF                             ; 1274
              04    AE        08    AE    9E 0000A              MOVAB   MASK_BUF+8, MASK_BUF+4                    ;
                                    6E    B4 0000F              CLRW    MASK_BUF                                  ; 1280
                                    56    D4 00011              CLRL    I                                         ; 1281
      25      08    AC              56    E1 00013 1$:          BBC     I, MASK_VECTOR, 5$                        ; 1282
                    0A              56    D1 00018              CMPL    I, #10                                    ; 1283
                                    05    13 0001B              BEQL    2$                                        ;
                    0B              56    D1 0001D              CMPL    I, #11                                    ;
                                    05    12 00020              BNEQ    3$                                        ;
                    57              04    D0 00022 2$:          MOVL    #4, BIT_NAME_LEN                          ;
                                    03    11 00025              BRB     4$                                        ;
                    57              03    D0 00027 3$:          MOVL    #3, BIT_NAME_LEN                          ;
                    50              6E    3C 0002A 4$:          MOVZWL  MASK_BUF, R0                              ; 1284
                    50        04    AE    C0 0002D              ADDL2   MASK_BUF+4, R0                            ;
                           0000'CF46    DF 00031              PUSHAL  BIT_NAME[I]                               ;
      60            9E              57    28 00036              MOVC3   BIT_NAME_LEN, @(SP)+, (R0)                ;
                    6E              57    A0 0003A              ADDW2   BIT_NAME_LEN, MASK_BUF                    ; 1285
                                    56    D6 0003D 5$:          INCL    I                                         ; 1281
```

```
                              0F          56  D1 0003F          CMPL    I, #15                          ;
                                          CF  1B 00042          BLEQU   1$                              ;
                                          6E  B5 00044          TSTW    MASK_BUF                        ; 1292
                                          02  13 00046          BEQL    6$                              ;
                                          6E  B7 00048          DECW    MASK_BUF                        ; 1293
                                          5E  DD 0004A 6$:      PUSHL   SP                              ; 1297
                               00000000G  8F  DD 0004C          PUSHL   #ANLOBJ$_MASK                   ;
                                      04  AC  DD 00052          PUSHL   INDENT_LEVEL                    ;
                                          7E  D4 00055          CLRL    -(SP)                           ;
                         FCF0  CF         04  FB 00057          CALLS   #4, ANL$FORMAT_LINE             ;
                                              04 0005C          RET                                     ; 1301

; Routine Size:  93 bytes,    Routine Base:  $CODE$ + 0412
```

```
%sbttl 'ANL$FORMAT_PROTECTION - Format Memory Protection Code'
!++
! Functional Description:
!       This routine is responsible for formatting a 4-bit memory
!       protection code in a nice way.
!
! Formal Parameters:
!       indent_level    The level of indentation for the line.
!       prot_code       The 4-bit protection code.
!
! Implicit Inputs:
!       global data
!
! Implicit Outputs:
!       global data
!
! Returned Value:
!       none
!
! Side Effects:
!
!--


global routine anl$format_protection(indent_level,prot_code): novalue = begin

own
        prot_code_table: vector[16,long] initial(
                            uplit byte (%ascic 'NA'),
                            uplit byte (%ascic '???'),
                            uplit byte (%ascic 'KW'),
                            uplit byte (%ascic 'KR'),
                            uplit byte (%ascic 'UW'),
                            uplit byte (%ascic 'EW'),
                            uplit byte (%ascic 'ERKW'),
                            uplit byte (%ascic 'ER'),
                            uplit byte (%ascic 'SW'),
                            uplit byte (%ascic 'SREW'),
                            uplit byte (%ascic 'SRKW'),
                            uplit byte (%ascic 'SR'),
                            uplit byte (%ascic 'URSW'),
                            uplit byte (%ascic 'UREW'),
                            uplit byte (%ascic 'URKW'),
                            uplit byte (%ascic 'UR'));

! Simply print a line with the protection code.

anl$format_line(0,.indent_level,anlobj$_protection,.prot_code_table[.prot_code]);

return;

end;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2
```

```
                                           41  4E  02  000A2 P.ABQ:  .ASCII  <2>\NA\                                        ;
                                       3F  3F  3F  03  000A5 P.ABR:  .ASCII  <3>\???\                                       ;
                                           57  4B  02  000A9 P.ABS:  .ASCII  <2>\KW\                                        ;
                                           52  4B  02  000AC P.ABT:  .ASCII  <2>\KR\                                        ;
                                           57  55  02  000AF P.ABU:  .ASCII  <2>\UW\                                        ;
                                           57  45  02  000B2 P.ABV:  .ASCII  <2>\EW\                                        ;
                                   57  4B  52  45  04  000B5 P.ABW:  .ASCII  <4>\ERKW\                                      ;
                                           52  45  02  000BA P.ABX:  .ASCII  <2>\ER\                                        ;
                                           57  53  02  000BD P.ABY:  .ASCII  <2>\SW\                                        ;
                                   57  45  52  53  04  000C0 P.ABZ:  .ASCII  <4>\SREW\                                      ;
                                   57  4B  52  53  04  000C5 P.ACA:  .ASCII  <4>\SRKW\                                      ;
                                           52  53  02  000CA P.ACB:  .ASCII  <2>\SR\                                        ;
                                   57  53  52  55  04  000CD P.ACC:  .ASCII  <4>\URSW\                                      ;
                                   57  45  52  55  04  000D2 P.ACD:  .ASCII  <4>\UREW\                                      ;
                                   57  4B  52  55  04  000D7 P.ACE:  .ASCII  <4>\URKW\                                      ;
                                           52  55  02  000DC P.ACF:  .ASCII  <2>\UR\                                        ;

                                                                     .PSECT  $OWN$,NOEXE,2

00000000' 00000000' 00000000' 00000000' 00000000' 00000000'  0037C PROT_CODE_TABLE:
                                                                     .ADDRESS P.ABQ, P.ABR, P.ABS, P.ABT, P.ABU, -
00000000' 00000000' 00000000' 00000000' 00000000' 00000000'  00394            P.ABV, P.ABW, P.ABX, P.ABY, P.ABZ, P.ACA, -  ;
                    00000000' 00000000' 00000000' 00000000'  003AC            P.ACB, P.ACC, P.ACD, P.ACE, P.ACF            ;



                                                                     .PSECT  $CODE$,NOWRT,2

                                                  0000 00000         .ENTRY  ANL$FORMAT_PROTECTION, Save nothing          ; 1326
                                50        08  AC  D0 00002           MOVL    PROT_CODE, R0                                ; 1350
                                     0000'CF40    DD 00006           PUSHL   PROT_CODE_TABLE[R0]                          ;
                                  00000000G  8F   DD 0000B           PUSHL   #ANLOBJ$_PROTECTION                          ;
                                          04  AC  DD 00011           PUSHL   INDENT_LEVEL                                 ;
                                              7E  D4 00014           CLRL    -(SP)                                        ;
                        FCD4    CF            04  FB 00016           CALLS   #4, ANL$FORMAT_LINE                          ;
                                                  04 0001B           RET                                                  ; 1354

; Routine Size:  28 bytes,    Routine Base:  $CODE$ + 046F
```

```
      1 %sbttl 'ANL$FORMAT_SEVERITY - Format Error Severity Code'
      1 !++
      1 ! Functional Description:
      1 !       This routine is called to format a standard VMS error severity
      1 !       code.  It also checks to make sure the code is valid.
      1 !
      1 ! Formal Parameters:
      1 !       indent_level    Level of indentation for report.
      1 !       severity        The severity code.
      1 !
      1 ! Implicit Inputs:
      1 !       global data
      1 !
      1 ! Implicit Outputs:
      1 !       global data
      1 !
      1 ! Returned Value:
      1 !       none
      1 !
      1 ! Side Effects:
      1 !
      1 !--
      1
      1
      2 global routine anl$format_severity(indent_level,severity): novalue = begin
      2
      2 own
      2         severity_code_table: vector[8,long] initial(
      2                                 uplit byte(%ascic 'WARNING'),
      2                                 uplit byte(%ascic 'SUCCESS'),
      2                                 uplit byte(%ascic 'ERROR'),
      2                                 uplit byte(%ascic 'INFO'),
      2                                 uplit byte(%ascic 'SEVERE'),
      2                                 uplit byte(%ascic 'reserved'),
      2                                 uplit byte(%ascic 'reserved'),
      2                                 uplit byte(%ascic 'reserved'));
      2
      2
      2 ! Format a line with the severity code on it.
      2
      2 anl$format_line(0,.indent_level,anlobj$_severity,.severity_code_table[.severity]);
      2
      2 ! Check for a reserved severity.
      2
      2 if .severity gequ 5 then
      2         anl$format_error(anlobj$_badseverity,.severity);
      2
      2 return;
      2
      1 end;


                                                               .PSECT  $PLIT$,NOWRT,NOEXE,2

             47  4E  49  4E  52  41  57  07  000DF P.ACG:  .ASCII  <7>\WARNING\                          ;
             53  53  45  43  43  55  53  07  000E7 P.ACH:  .ASCII  <7>\SUCCESS\                          ;
```

```
                              52  4F  52  52  45  05  000EF P.ACI:   .ASCII  <5>\ERROR\                                      ;
                                  4F  46  4E  49  04  000F5 P.ACJ:   .ASCII  <4>\INFO\                                       ;
                          45  52  45  56  45  53  06  000FA P.ACK:   .ASCII  <6>\SEVERE\                                     ;
              64  65  76  72  65  73  65  72  08  00101 P.ACL:   .ASCII  <8>\reserved\                                   ;
              64  65  76  72  65  73  65  72  08  0010A P.ACM:   .ASCII  <8>\reserved\                                   ;
              64  65  76  72  65  73  65  72  08  00113 P.ACN:   .ASCII  <8>\reserved\                                   ;

                                                                 .PSECT  $OWN$,NOEXE,2

00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 003BC SEVERITY_CODE_TABLE:
                                                                 .ADDRESS P.ACG, P.ACH, P.ACI, P.ACJ, P.ACK, -       ;
                                          00000000' 00000000' 003D4         P.ACL, P.ACM, P.ACN                         ;


                                                                 .PSECT  $CODE$,NOWRT,2

                                                    0004 00000          .ENTRY  ANL$FORMAT_SEVERITY, Save R2            ; 1379
                          52       08   AC   D0 00002          MOVL    SEVERITY, R2                                    ; 1395
                                 0000'CF42   DD 00006          PUSHL   SEVERITY_CODE_TABLE[R2]                         ;
                             00000000G  8F   DD 0000B          PUSHL   #ANLOBJ$_SEVERITY                               ;
                                   04   AC   DD 00011          PUSHL   INDENT_LEVEL                                    ;
                                        7E   D4 00014          CLRL    -(SP)                                           ;
                     FCB8   CF          04   FB 00016          CALLS   #4, ANL$FORMAT_LINE                             ;
                          05            52   D1 0001B          CMPL    R2, #5                                          ; 1399
                                        0D   1F 0001E          BLSSU   1$                                              ;
                                        52   DD 00020          PUSHL   R2                                              ; 1400
                             00000000G  8F   DD 00022          PUSHL   #ANLOBJ$_BADSEVERITY                            ;
                     FD2B   CF          02   FB 00028          CALLS   #2, ANL$FORMAT_ERROR                            ;
                                        04   0002D 1$:         RET                                                     ; 1404

; Routine Size:  46 bytes,    Routine Base:  $CODE$ + 048B
;  985        1405  1 %sbttl 'ANL$INTERACT - See If User Wants to Continue'
;  986        1406  1 !++
;  987        1407  1 ! Functional Description:
;  988        1408  1 !       This routine is called as part of the processing of the /INTERACTIVE
;  989        1409  1 !       qualifier.  We see if the user wants to continue with this file,
;  990        1410  1 !       or quit.
;  991        1411  1 !
;  992        1412  1 ! Formal Parameters:
;  993        1413  1 !       none
;  994        1414  1 !
;  995        1415  1 ! Implicit Inputs:
;  996        1416  1 !       global data
;  997        1417  1 !
;  998        1418  1 ! Implicit Outputs:
;  999        1419  1 !       global data
; 1000        1420  1 !
; 1001        1421  1 ! Returned Value:
; 1002        1422  1 !       True if user wants to continue; false otherwise.
; 1003        1423  1 !
; 1004        1424  1 ! Side Effects:
; 1005        1425  1 !
; 1006        1426  1 !--
; 1007        1427  1
; 1008        1428  1
; 1009        1429  2 global routine anl$interact = begin
; 1010        1430  2
; 1011        1431  2 local
; 1012        1432  2         status: long,
; 1013        1433  2         local_described_buffer(answer_buf,1);
; 1014        1434  2
; 1015        1435  2
; 1016        1436  2 ! First we display a message telling the user what to do.
; 1017        1437  2
; 1018        1438  2 anl$format_line(-1,0,anlobj$_interact);
; 1019        1439  2
; 1020        1440  2 ! Now we get the user's answer.  If it is a period (.), then we return
; 1021        1441  2 ! false.  If it's blank, we return true.  If CTRL/Z, we just bag it.
; 1022        1442  2
; 1023        1443  2 status = lib$get_input(answer_buf);
; 1024        1444  2 if .status eqlu rms$_eof then
; 1025        1445  2         anl$exit_with_status();
; 1026        1446  2 return ch$rchar(.answer_buf[ptr]) nequ '.';
; 1027        1447  2
; 1028        1448  1 end;



                                  0000 00000          .ENTRY  ANL$INTERACT, Save nothing                   ; 1429
              5E                08  C2 00002          SUBL2   #8, SP                                       ;
                                01  DD 00005          PUSHL   #1                                           ; 1433
       04     AE        08      AE  9E 00007          MOVAB   ANSWER_BUF+8, ANSWER_BUF+4                   ;
                    00000000G   8F  DD 0000C          PUSHL   #ANLOBJ$_INTERACT                            ; 1438
                                7E  D4 00012          CLRL    -(SP)                                        ;
              7E                01  CE 00014          MNEGL   #1, -(SP)                                    ;
     FC89     CF                03  FB 00017          CALLS   #3, ANL$FORMAT_LINE                          ;
```

```
                                          5E  DD 0001C          PUSHL   SP                                          ; 1443
                   00000000G  00          01  FB 0001E          CALLS   #1, LIB$GET_INPUT                           ;
                   0001827A   8F          50  D1 00025          CMPL    STATUS, #98938                              ; 1444
                                          05  12 0002C          BNEQ    1$                                          ;
                       FE00   CF          00  FB 0002E          CALLS   #0, ANL$EXIT_WITH_STATUS                    ; 1445
                                          50  D4 00033  1$:     CLRL    R0                                          ; 1446
                         2E         04    BE  91 00035          CMPB    @ANSWER_BUF+4, #46                          ;
                                          02  13 00039          BEQL    2$                                          ;
                                          50  D6 0003B          INCL    R0                                          ;
                                              04 0003D  2$:     RET                                                 ; 1448

; Routine Size:  62 bytes,    Routine Base:  $CODE$ + 04B9


; 1029          1449  1
; 1030          1450  0 end eludom



                                                                        .EXTRN  LIB$SIGNAL

;                                  PSECT SUMMARY
;
;       Name                       Bytes                        Attributes
;
;  $GLOBAL$                            4  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                             988  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                            284  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                           1271  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                               Library Statistics
;
;                                              -------- Symbols --------        Pages       Processing
;       File                                   Total     Loaded   Percent       Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1            9776         50         0         581         00:01.0




;                                   COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:OBJEXEOUT/OBJ=OBJ$:OBJEXEOUT MSRC$:OBJEXEOUT/UPDATE=(ENH$:OBJEXEOUT)

; Size:           1271 code + 1276 data bytes
; Run Time:          00:28.7
; Elapsed Time:      00:54.1
; Lines/CPU Min:       3028
; Lexemes/CPU-Min:  20190
; Memory Used:   187 pages
; Compilation Complete
```

0006 AH-BT13A-SE DIGITAL EQUIPMENT CORPORATION
VAX/VMS V4.0 CONFIDENTIAL AND PROPRIETARY